Construction of Irregular LDPC Codes by Row Partition

Xiongfei Tao, Yan Zhang, Pan Liu, and Zuoqi Hu

Abstract In this paper, we propose an approach to construct a class of irregular low-density parity-check (LDPC) codes based on row partition. We divide the rows of LDPC check matrix into subsets and derive an approach to arrange '1' in each column to prevent low weight code words. When decoded by iterative algorithm, the proposed codes show performance with low error floor and are subject to few undetected errors. The LDPC codes constructed based on the proposed scheme have efficiently encoding structures.

KeyWords LDPC codes, minimum distance, row partition.

1 Introduction

Low-density parity-check (LDPC) codes were original invented by Gallager in 1963 [1], he considered only regular LDPC codes. Luby and etc[2] introduced irregular graph into LDPC codes, which improved the performance of the LDPC codes in waterfall region. Richardson[3] employed density evolution (DE) to predict the threshold of LDPC codes with a certain degree profile, moreover density evolution can be used to optimize degree profile with a threshold near Shannon limit. However, an LDPC code with an optimized degree profile does not necessarily guarantee good performance of the code.

The performance of an LDPC code depends on not only the structure of its Tanner graph and its weight distribution of code words. Originally, most construction

Xiongfei Tao(✉)
School of Optical and Electronics Information, Huazhong University of Science and Technology
Wuhan, Hubei, P.R.China
taoxiongfei@mail.hust.edu.cn

Yan Zhang, Pan Liu, and Zuoqi Hu
School of Optical and Electronics Information, Huazhong University of Science and Technology
Wuhan, Hubei, P.R.China

This work has been supported by the National Science Foundation of China (No. 60902006) and the Scientific Research Foundation of Huazhong University of Science and Technology (No.2012QN152).

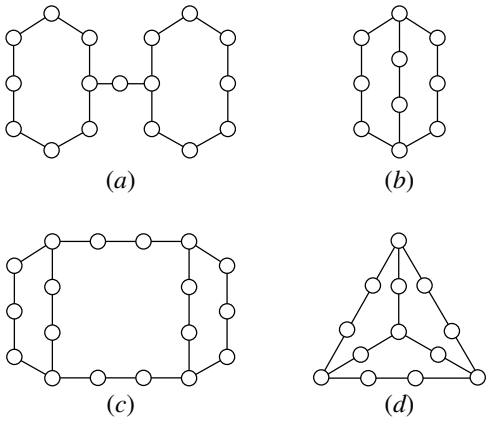

Figure 1: Graphs of low weight code words structure

methods focused on constructing LDPC codes with big minimum cycle (or girth) which is suitable for iterative decoding. An well-known method to construct LDPC codes with large girth is progressive edge growth (PEG)[4] algorithm. Although the girth and minimum distance of an LDPC code have relationship to some extent, the girth does not guarantee large minimum distance. The LDPC codes with large girth may still have low weight code words, especially irregular codes.

Approaches to the construction of LDPC codes can divide into two major classes: random-like constructions and algebraic constructions. Random approaches can construct LDPC codes that closely achieve the Shannon capacity but the significant encoding complexity of random codes leads to hard implementation in hardware while structured LDPC codes have hardware-friendly encoding. However, even well designed LDPC codes have large minimum distance, their code words weight distribution may not be good. Take QC-LDPC codes as an example, QC-LDPC codes are popular codes in nowadays. An QC-LDPC code is consist of numbers of subblocks, each subblock is either a $p \times p$ zero matrix or a $p \times p$ circulant shift matrix of an identity matrix. Due to the quasi cyclic character of QC-LDPC codes, the number of a certain code word weight is multiplicity of p. The presence of a large number of low-weight codewords is critical to both the error floor and probability of undetected errors.

In this paper, a design method of the irregular LDPC codes is proposed to obtain both good performance and practically allowable encoding complexity.

2 Low weight code words of irregular LDPC codes

The target of most methods on the construction of irregular LDPC codes is to design code graphs with good properties, such as girth. However, irregular LDPC codes

have a large amount of bit nodes with degree 2 and 3. Large cycle does not guarantee large distance. As a typical example, suppose that there is a cycle which consist of d bit nodes with degree 2. The module-2 sum of the columns corresponding to these bit nodes is a zero vector, hence there is a code word with weight d. Although this condition can be broken by making the degree 2 bit columns cycle free among them, there exists other condition of low weight code words constituted of some degree 2 and a small amount of degree 3 bit nodes, and according to the property of code words of linear codes, the multiplicity of degree 3 bit nodes is even. Some examples are shown in Fig.1. Fig.1(a) and (b) illustrate how low weight code words consist of some bit nodes with degree 2 and two bit nodes with degree 3, and Fig.1(c) and (d) give examples of low weight code words constituted of some bit nodes with degree 2 and four bit nodes with degree 3.We translate the subgraph in Fig.1(a) and (b) into the form of matrix as shown in Eq.(2) and Eq.(1).

$$H_d = \left(\begin{array}{ccc|ccc|ccc|c} 1 & & & & & & & & & 1 \\ 1 & \ddots & & & & & & & & \\ & \ddots & 1 & & & & & & & \\ & & 1 & & & & & & & 1 \\ \hline & & & 1 & & & & & & 1 \\ & & & 1 & \ddots & & & & & \\ & & & & \ddots & 1 & & & & \\ & & & & & 1 & & & & 1 \\ \hline & & & & & & 1 & & & 1 \\ & & & & & & 1 & \ddots & & \\ & & & & & & & \ddots & 1 & \\ & & & & & & & & 1 & 1 \end{array}\right) \tag{1}$$

$$H_d = \left(\begin{array}{ccc|ccc|ccc|c} 1 & & & & & & & & & 1 \\ 1 & \ddots & & & & & & & & \\ & \ddots & 1 & & & & & & & \\ & & 1 & & & & & & & 1 \\ \hline & & & 1 & & & & & & 1 \\ & & & 1 & \ddots & & & & & \\ & & & & \ddots & 1 & & & & \\ & & & & & 1 & & & & 1 \\ \hline & & & & & & 1 & & & 1 \\ & & & & & & 1 & \ddots & & \\ & & & & & & & \ddots & 1 & \\ & & & & & & & & 1 & 1 \end{array}\right) \tag{2}$$

It is shown in the matrices that the degree 2 bit nodes are divided into three groups and there are two degree 3 bit nodes in those matrices. Each columns with degree 3 in Eq.(2) has 2 '1' in the same group, and the remaining '1' in the two columns are in a common group. For Eq.(1), '1's in the 2 columns with degree 3 are in the same 3 groups. In Fig.1(c) and (d), the code words consist of several degree 2 bit nodes and 4 degree 3 bit nodes and there exists cycles constitute of several degree 2 bit nodes and 3 or 4 degree 3 bit nodes.

3 The approach of our construction

3.1 Method to eliminate low weight code words

For the construction of an LDPC code whose check matrix has m rows and n columns, We divide the m rows of the check matrix into g partition, we put columns with weight 2 in the parity-check position of each partition and there are no columns with weight 2 crossover any two partitions. We make weight 2 columns cycle free, the cycle free condition can be achieved by downward shifting two consecutive '1' from the top of each partition until the second '1' reaches the bottom of them, hence, there are no code words consist of only degree 2 bit nodes. To eliminant the code words consist of several degree 2 bit nodes and 2 degree 3 bit nodes, we adopt the following principle:

- If there are 2 columns in both of which 2 '1's locate in the same partition, then the two remaining '1's in the columns should not share a common partition;
- For any pair columns in which 3 '1's are in different partitions, then the pair columns should share at most two partitions.

For the low weight code words in other cases like that of Fig.1(c) and (d), in which there are four degree 3 bit nodes, it is nearly impossible to eliminant the pattern if the construction of middle length codes are considered. But as mentioned before, there exists cycles constitute of several degree 2 bit nodes and 3 or 4 degree 3 bit nodes in such pattern of code words, thus if we enlarge cycles in such cases, the weight of the code words in which 4 degree 3 bit nodes participate should be improved potentially. Consider a column with even weight, the '1's in this column should possess several partitions, if there are even '1's in each partition then it is a big chance to form a low weight code word, this condition should be avoided during the construction. The whole procedure is built on standard PEG algorithm. After a tree expansion is obtained, we try to find an appropriate partitions which prevent the forming of low weight code words. Moreover when expanding a tree from a bit node, if there are already 4 degree 3 bit nodes in a branch then we do not add degree 3 bit nodes into this branch anymore. This tactics maximize the the cycles constitute of several degree 2 bit nodes and 3 or 4 degree 3 bit nodes.

3.2 A fast encoding method

H refers to the check matrix and c refers to the codeword. H is divided into two parts H_p and H_u, and H_p is further divided into H_{p1} and H_{p2},

$$H = (H_p|H_u) = (H_{p1}|H_{p2}|H_u) \tag{3}$$

where H_{p1} and H_{p2} are corresponding to p_1 and p_2 which are redundant bit nodes of c and H_u is corresponding to u which are information bit nodes of c. H_{p1} represent the $m-g$ columns with weight 2, we put g columns with weight 3 in H_{p2}, and arrange the '1's based on the following principle:

- For the first column, we put a '1' in the first row of the first partition, and put two '1's in the first and last row of the next partition;
- The other columns can be obtain by downward cyclic shifting the pattern of the first column by $g-1$ times.

Under this design, there are only two non-zero columns in each partition of H_{p2}, the weights of these 2 columns are 1 and 2 respectively. For each partition, we make row transformation by adding other rows to the first one, then only one '1' remain in the first row of each partition in H_p. By performing column permutations, H_p will be in the form of upper triangle, which leads the row rank of the check matrix equal to m, and hence its null space gives an (n, m) LDPC code with an actual rate of 1-m/n.

We can easily solve the first equation in each partition and get the value of p_2 and the value of p_1 can be calculate by backward substitution. And furthermore, after the value of p_2 is obtained, the equations in one partition are independent of the equations in other partition, hence each partition can be encoded in parallel.

4 Simulation Result

In this part, we estimate the performance of (1728, 864) irregular LDPC codes constructed based on PEG algorithm, Jinhu's[5] approach and our approach over an additive white Gaussian noise (AWGN) channel, with the standard BP decoding and a maximum number I_{max}=100 iterations. The bit and check degree distributions of the PEG-based code and our code are $\lambda(x) = 0.2716x + 0.3333x^2 + 0.3951x^7$ and $\rho(x) = 0.2222x^5 + 0.7778x^6$, respectively. In Jinhu's paper, he changed the distributions which is suitable for his construction. Under this distributions, there are 792, 648 and 288 bit nodes with degree 2, 3 and 8 respectively. We divide the 864 rows of the check matrix into 72 partitions, there are 12 rows in each partition.

We apply the impulse method of [6] to compute low-weight codewords for the three (1728, 864) codes and list the weight distributions in Table 1. From Table 1, we observe that the minimum distances of PEG code, Jinhu's code and proposed code

Table 1: List of Weight Profiles For Three Codes

w	PEG	Jinhu	pro.	w	PEG	Jinhu	pro.
14	1	-	-	20	5	36	2
15	-	-	-	21	6	-	5
16	1	-	-	22	8	-	3
17	2	-	1	23	10	2	11
18	1	36	3	24	22	37	15
19	3	-	2	25	29	36	23

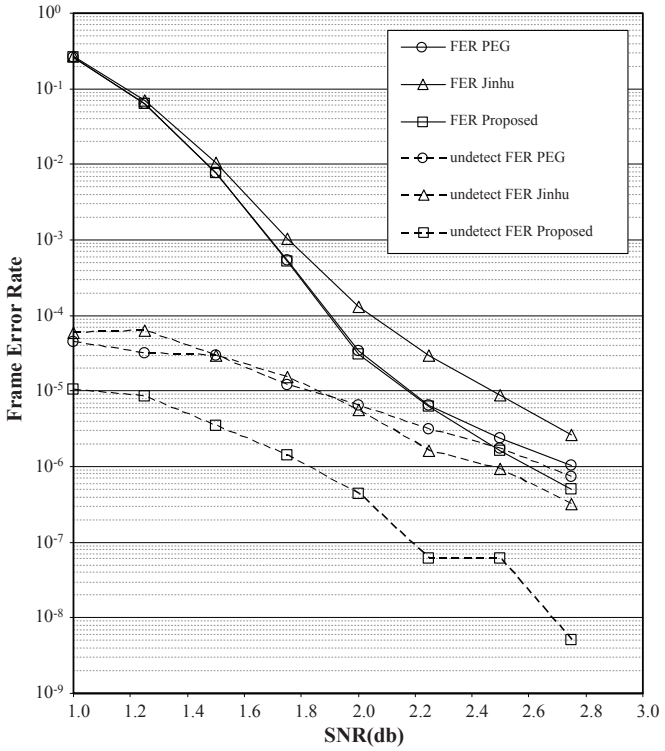

Figure 2: Simulation result and performance comparison

are 14, 18 and 17 respectively. Although the minimum distance of proposed code is slightly smaller than Jinhu's code, the proposed code shows a better distribution of code weight. We also test the girth parameters of the three codes, the girths of these codes are 8, 4 and 8 respectively.

Fig.2 depicts the frame error rate (FER) and undetected frame error rate of these three codes. We observe that in the high signal-to-noise ratio (SNR) region, the proposed scheme reduces the error floor compared to the PEG-based LDPC code.

We also observe that the LDPC code constructed based on the proposed approach has much less undetected errors than the PEG-based LDPC code. The undetected FER of the proposed code is about two orders of magnitude below that of the PEG one. Fig.2 also shows that the proposed code has better performance than Jinhu's code. Although the minimum distance of Jinhu's code is the largest among these three codes, but we believe that it is the poor weight distribution that leads to relative worse performance of FER and undetected FER.

5 Conclusion

In this paper, a construction of irregular LDPC codes is proposed. To design a irregular LDPC code with good distance distribution, we divide the check matrix into several partitions, in each partition columns with weight 2 is cycle free, and any two partitions is not linked by columns with weight 2. We select location of '1's in each columns to avoid the code words consist of several degree 2 bit nodes and 2 degree 3 bit nodes and avoid the low weight codes with other patterns by enlarge the cycles consist of several degree 2 bit nodes and 3 or 4 degree 3 bit nodes. We derive a structure of such irregular LDPC codes to achieve fast encoding.

References

[1] R. G. Gallager, "Low density parity check codes", *IEEE Trans. Inform. Theory*, vol.IT-8, no.1, pp.21-28, Jan. 1962.

[2] M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D.A.Spielman, Improved low-dertsity parity-check codes using irregular graphs, IEEE Trans. Inform. Theory, vol. 47, pp. 585-598, Feb. 2001.

[3] T. J. Richardson and R. L. Urbanke, The capacity of low-density parity-check codes under message-passing decoding, IEEE Trans. Inf. Theory, vol. 47, no. 2, pp. 599C618, Feb. 2001.

[4] X.-Y. Hu, R.M. Tanner, J.-T. Zhang and M.P.C. Fossorier, Construction of Irregular LDPC Codes by Quasi-Cyclic Extension, IEEE Trans. Inf. Theory, vol. 53, no. 4, pp. 1479 - 1483, APR. 2007. pp. 995C1001.

[5] J.-H. Chen, E. Eleftheriou, and D.-M. Arnold, Progressive edge-growth tanner graphs, in Proc. Globecom 2001, San Antonio, TX, Nov. 2001, pp. 995C1001.

[6] X. Hu, M. Fossorier, and E. Eleftheriou, On the computation of the minimum distance of low-density parity-check codes, in Proc. IEEE Int. Conf. Communications, Paris, France, Jun. 2004, pp. 767C771.